WHAT IS CLAIMED

1. Interfaces, embodied on one or more computer-readable media, to be called on kernel transaction management objects, comprising:

application program interfaces (APIs) to implement operations on a kernel transaction object (TX);

APIs to implement operations on a kernel resource management object (RMO); and

APIs to implement operations a kernel enlistment (EN) object.

2. Interfaces according to Claim 1, wherein each of the APIs to implement operations on TX, RMO, and EN utilize a handle to refer to an object.

3. Interfaces according to Claim 2, wherein each of the handles is an opaque reference to a unique object.

4. Interfaces according to Claim 2, wherein at least one API calls for TX to transmit pre-prepare messages to resource managers associated with a transaction.

5. Interfaces according to Claim 2, wherein at least one API calls for TX to transmit a prepare request to resource managers enlisted in a transaction.

6. Interfaces according to Claim 2, wherein at least one API calls for a new TX to be created for a transaction.

7. Interfaces according to Claim 2, wherein at least one API calls for an existing TX to be opened for a transaction.

8. Interfaces according to Claim 2, wherein at least one API calls for TX to commit a transaction.

9. Interfaces according to Claim 2, wherein at least one API calls for TX to abort a transaction.

10. Interfaces according to Claim 2, wherein at least one API calls for TX to save a current state of the transaction.

11. Interfaces according to Claim 2, wherein at least one API calls for TX to retrieve information about the TX for a requestor.

12. Interfaces according to Claim 2, wherein at least one API calls for TX to set information.

13. Interfaces according to Claim 2, wherein at least one API calls for TX to close.

14. Interfaces according to Claim 2, wherein at least one API is:

PreprepareEnlistment,

PrepareEnlistment,

OpenEnlistment

CreateTransaction,

OpenTransaction,

CommitTransaction,

RollbackTransaction,

SavepointTransaction,

GetTransactionInfo, and

SetTransactionInfo.

15. Interfaces according to Claim 2, wherein at least one API calls for a new RMO to be created.

16. Interfaces according to Claim 15, wherein the new RMO is volatile.

17. Interfaces according to Claim 15, wherein the new RMO is durable.

18. Interfaces according to Claim 2, wherein at least one API calls for an existing RMO to open for a transaction.

19. Interfaces according to Claim 2, wherein at least one API calls for RMO to be destroyed.

20. Interfaces according to Claim 2, wherein at least one API calls for RMO to transmit information regarding RMO to a requestor.

21. Interfaces according to Claim 2, wherein at least one API calls for RMO to set information.

22. Interfaces according to Claim 2, wherein at least one API calls for RMO to be enlisted on a transaction at least once.

23. Interfaces according to Claim 2, wherein at least one API calls for a notification from a resource manager for RMO.

24. Interfaces according to Claim 2, wherein at least one API is:
CreateResourceManager,
OpenResourceManager,
DestroyResourceManager,
GetResourceManagerInfo,
SetResourceManagerInfo,
CreateEnlistment, and
GetNotificationResourceManager.

25. Interfaces according to Claim 2, wherein at least one API is to implement operations on TX by RMO.

26. Interfaces according to Claim 25, wherein the at least one API is to inform TX that pre-preparing is complete.

27. Interfaces according to Claim 25, wherein the at least one API is to inform TX that transaction preparation has been completed by a requested resource manager.

28. Interfaces according to Claim 25, wherein the at least one API is to inform TX that a resource manager has completed rolling back a transaction.

29. Interfaces according to Claim 25, wherein the at least one API is to inform TX that a resource manager has committed to a transaction.

30. Interfaces according to Claim 25, wherein the at least one API is:

PrePrepareComplete,

PrepareComplete,

RollbackComplete, and

CommitComplete.

31. Interfaces according to Claim 2, wherein at least one API calls for a resource manager to be registered as a communications resource manager for a particular protocol.

32. Interfaces according to Claim 2, wherein at least one API calls for a representation of a transaction to be serialized into a buffer.

33. Interfaces according to Claim 2, wherein at least one API calls for information representing registered protocols to be serialized into a buffer.

34. Interfaces according to Claim 32, wherein at least one API calls for a transaction represented by the serialization be made available by a transaction management object.

35. Interfaces according to Claim 2, wherein at least one API calls for a transaction to be propagated to a destination using push-style propagation.

36. Interfaces according to Claim 35, wherein at least one API calls for the output of the API calls for the transaction to be propagated to a destination using push-style propagation to be retrieved.

37. Interfaces according to Claim 31, wherein at least one API is called when transaction propagation has been completed.

38. Interfaces according to Claim 31, wherein at least one API is called when requested transaction propagation has failed.

39. Interfaces according to Claim 2, wherein at least one API is:

RegisterProtocolAddressInformation,

MarshallTransaction,

GetProtocolAddressInformation,

PullTransaction,

PushTransaction,

PushTransactionBuffer,

PropagationComplete, and

PropagationFailed.

40. An apparatus for implementing a transaction, comprising:

a kernel transaction object (TX);

a kernel resource manager object (RMO); and

a kernel enlistment object (EN),

wherein two-phase commit processing is executed by calling APIs on the TX, the RMO, and the EN.

41. A transaction method for a kernel transaction object, comprising:

receiving a call to open;

transmitting a call to prepare for a transaction;

receiving a call confirming prepare complete;

transmitting a call indicating an outcome of the transaction; and

receiving a call confirming receipt of the call indicating the outcome of the transaction.

42. A method according to Claim 41, wherein the call to open is received from a client application.

43. A method according to Claim 41, wherein the call to prepare is transmitted to resource managers enlisted on a transaction.

44. A method according to Claim 43, wherein the call to prepare supplies a handle for the transaction to be prepared for, and supplies a handle to the kernel transaction object.

45. A method according to Claim 41, wherein the call confirming prepare complete is received from a resource manager enlisted on the transaction.

46. A method according to Claim 45, wherein the call confirming prepare complete includes a handle indicating the transaction for which the prepare operation has been completed.

47. A method according to Claim 41, wherein the call indicating the outcome of the transaction is transmitted to a resource manager enlisted on the transaction.

48. A method according to Claim 41, further comprising receiving a call to be committed to the transaction.

49. A method according to Claim 48, wherein the call to be committed to the transaction supplies a pointer to a location that will receive a handle to the

transaction, supplies a mask specifying a desired level of access, and supplies a pointer to an optional object attribute structure.

50. A method according to Claim 41, further comprising receiving a call to abort the transaction.

51. A method according to Claim 50, wherein the call to abort the transaction supplies a handle indicating the transaction to be aborted.

52. A transaction method for a kernel resource management object, comprising:

receiving a call to prepare a resource for a transaction;

transmitting a call confirming prepare complete;

receiving a call indicating an outcome of the transaction; and

transmitting a call confirming receipt of the call indicating the outcome of the transaction.

53. A method according to Claim 52, wherein the call to prepare is received from a transaction object.

54. A method according to Claim 53, wherein the call to prepare includes a handle for the transaction to be prepared for, and supplies a handle to the transaction object.

55. A method according to Claim 53, wherein the call confirming prepare complete is transmitted to a transaction object.

56. A method according to Claim 55, wherein the call confirming prepare complete supplies a handle indicating the transaction for which the prepare operation has been completed.

57. A method according to Claim 52, wherein the call indicating the outcome of the transaction is received from a transaction object.

58. A method according to Claim 52, further comprising transmitting a call indicating the transaction has been aborted.

59. A method according to Claim 58, wherein the call indicating the transaction has been aborted is transmitted to a transaction object, and supplies a handle indicating the transaction for which the abort has been completed.

60. A method according to Claim 52, further comprising transmitting a call indicating the transaction has been committed.

61. A method according to Claim 60, wherein the call indicating the transaction has been committed is transmitted to a transaction object, and supplies a handle indicating the transaction for which the commit has been completed.

62. A method according to Claim 52, further comprising setting resource data in accordance with the outcome of the transaction.

63. A computer-readable medium on which are stored one or more functions, calls for the one or more functions causing a kernel transaction object to:

open a transaction object;

transmit a prepare call;

receive a prepare complete call;

transmit a call indicating an outcome of the transaction; and

receive a call indicating receipt of the call indicating the outcome of the transaction.

64. A computer-readable medium according to Claim 63, wherein the call to open the transaction object is received from a client application.

65. A computer-readable medium according to Claim 63, wherein the prepare call is transmitted to resource managers enlisted on a transaction.

66. A computer-readable medium according to Claim 63, wherein the call confirming prepare complete is received from a resource manager enlisted on the transaction.

67. A computer-readable medium according to Claim 63, wherein the call indicating the outcome of the transaction is transmitted to a resource manager enlisted on the transaction.

68. A computer-readable medium according to Claim 63, further comprising a call causing the kernel transaction object to be committed to the transaction.

69. A computer-readable medium according to Claim 63, further comprising a call causing the kernel transaction object to abort the transaction.

70. A computer-readable medium on which are stored one or more functions, calls for the one or more functions causing a kernel resource management object to:

prepare a resource for a transaction;

transmit a call confirming prepare complete; and

set resource data in accordance with an outcome of the transaction.

71. A computer-readable medium according to Claim 70, wherein the call to prepare a resource for the transaction is received from a transaction object.

72. A computer-readable medium according to Claim 70, wherein the call confirming prepare complete is transmitted to a transaction object.

73. A computer-readable medium according to Claim 70, wherein a call indicating the outcome of the transaction is received from a transaction object.

74. An apparatus for implementing a transaction, comprising:

means for representing a transaction; and

means for representing a relationship between the means for representing a transaction and a resource participating in the transaction,

wherein two-phase commit processing is executed by calling APIs between the means for representing a transaction and the means for representing a relationship.